NTNU – Trondheim
Norwegian University of
Science and Technology

Title

Vegard Stjerna Lindrup

Title: Title

Student: Vegard Stjerna Lindrup

Problem description:

    Dette legges til i DAIM, og blir derfor fjernet før innlevering.

Responsible professor: Tor Engebret Onshus

Supervisor:

Abstract

Mobile robot platforms etc...

Sammendrag

Mobile robotplatformer kan kjøre rundt og...

Preface

Hva synes jeg om oppgaven? Kjempeartig! Eget acknowledgemet-kapittel?

Contents

List of Acronyms

AI Artificial Intelligence

CP Cathodic Protection

GUI Graphical User Interface

HMI Human-Machine Interaction

ISS International Space Station

LIDAR LIght Detection And Ranging

MIT Massachusetts Institute of Technology

NUI Natural user interface

NDT Non-destructive Testing

OCS Operator Control Station

PR Personal Robot

ROS Robot Operating System

RTAB-Map Real-Time Appearance-Based Mapping

SDF Simulation Description Format

SLAM Simultanious Localization And Mapping

STAIR Stanford AI Robot

URDF Unified Robot Description Format

List of Figures

List of Tables

List of Algorithms

Chapter 1
Introduction

Introduction

1.1 About the Thesis

1.2 Autonomous Mobile Robotic Maintenance

1.2.1 What and Why?

1.2.2 State of the art

1.2.3 Notable Projects

1.2.4 Future Goal (The final product)

A nice description of a potential final product.

1.2.5 State of the Art in Autonomous Robots

Notable projects etc.

1.3 Implementation Overview

1.4 Thesis Structure

Chapter 2

Background Theory

2.1 Robotic Maintenance of Industrial Installations

2.1.1 Introduction

This project is only a part of a larger long-term goal concerning robotic maintenance. The goal of this section is to put the following background theory, and the implementation described in chapter 3, into the context of the larger topic, which is automated robotic maintenance on industrial installations. It is important to describe how maintenance and inspection of industrial installations is done today, before application of robotic maintenance is discussed.

2.1.2 Potential Maintenance Tasks

Hidden failure modes: PFD: What is the probability that a device (Fire detector, shut down valve, etc.) will fail when needed? Solution: Periodic maintenance.

2.1.3 Offshore Installations

Corrosion

Offshore installations are regularly, if not continuously exposed to harsh weather conditions in the form of wind and seawater. Presence of seawater, either through direct contact or in the form of drops and vapour, forms a very corrosive environment. In addition, the offshore environment is classified as the most corrosive environment in ISO 12944[ER12]. Common corrosion prevention methods are[ER12]:

- Sacrificial Anodes.

- Cathodic Protection (CP) in the form of a DC-current.

- Protective coating.

In terms of maintenance, the sacrificial anodes can be subjected to periodic inspections and replacements, which could be done a robot. CP can more easily be implemented with automated self tests, and should normally not require any inspections and maintenance[ER12]. Application of protective coating should ideally be applied in the controlled environment of a workshop. If protective coating is to be applied at sea, one should strive to make the conditions as favourable as possible.

Fatigue

Waves, wind, water currents and other forces subject offshore installations to structural stress.

2.2   Robotic Maintenance Today

Gas leak detection: [SS14]. DARPA robotic challenge. Industrial ROS.

Today, robots outside of the R&D laboratories and universities are usually put to perform a very limited set of specific tasks. Today, modern robots outperform humans in specific, repetitive and tedious tasks in controlled environments. For robotics in uncontrolled environments, there are still much research and development to be done. Several recent research papers(CITE en HAUG MED PAPERS!) indicate that significant advances are being made in the labs universities around the globe. Nevertheless, it is still some time before industrial robots will expand beyond repetitive and tedious tasks in controlled environments on on a large scale[ifr]. A notable trend, and a step in the direction of uncontrolled environments, is increased human-robot collaboration.

A multitude of organizations and agents, e.g. *ROS - Industrial*[ROSb], Springer with the Springer Tracts in Advanced Robotics as well as the DARPA Robotics Challenge (DRC)[DRC], aims to incite innovation and progress through competition, information flow and collaboration.

that could probe useful in robotic maintenance are being developed a

2.3   Modelling and Simulation

2.3.1   Some Terminology

Coordinate Systems and Poses

Robot Joints

All links are connected to each other by joints.

Coordinate systems are essential in the field of robotics.

2.3.2   Robot Modelling

2.3.3   Simulating in Gazebo

2.4   ROS

2.4.1   Introduction

The Robot Operating System (ROS) is a collection of software libraries, tools and drivers intended for robot software development. A ROS installation can be taylored to meet the demands of a wide range of robots with variyng complexity. ROS is usually installed in the form of an already built Debian-package. These packages are only compatible with a few versions of Ubuntu which are specified on the ROS homepage. When installed and configured, ROS will run on top of Linux, and can in many be percieved as and extention of Linux itself. Installing ROS from source is possible, but not recomended [ROSc].

Roots of ROS can be traced back to Stanford University at the beginning of the 2000s. At Stanford, several robotics software frameworks, including Stanford AI Robot (STAIR) and the Personal Robot (PR) program, were created to provide dynamic, flexible and well tested foundations for further robot development and research. In 2007, a nearby start-up company and robot incubator, Willow Garage, sought to build upon these concepts, and initiated a collaborative and open development process of a new software framework, which eventually became ROS[ROSa][QGS15]. This framework can be used under the BSD open-source license, which means that ...[**?**] Today, ROS comes in many forms and comprise hundreds of advanced packages, algorithms and drivers, making it applicable for hobbyists, industrial automation and everything in between.

2.4.2   Important ROS Concepts

The following descriptions are included in order to provide a complete, self-contained description of the project implementation. Similar descriptions can be found on the official ROS website (INSERT LINK HERE), as well as in any book on ROS (for example [QGS15]).

The ROS Graph

A ROS system comprise a set of small programs that communicate with each other.

roscore

The *roscore* is an essensial part of any ROS system. It enables nodes to communicate with each other.

Topics

Services

rosbridge

title

2.4.3   ROS-Related Tools

Robot Modelling In URDF

Visialization in RVIZ

Simulation in Gazebo

2.4.4   Structure of a ROS Application

2.4.5   Notable Robots Running ROS

PR2 - Personal Robot 2   PR2 is one of the first robots designed to run ROS [QGS15], and also one of the most advanced and capable robots with ROS today.

TurtleBot   The TurtleBot is a cheaper ROS-ready alternative to the PR2.

Robonaut 2   Robonaut 2, a dexterous humanoid robot, currently resides within the International Space Station (ISS) 400 km above the earth's surface. In 2014, a SpaceX Dracon capsule brought ROS as well as a pair of legs for Robonaut up to the ISS[**?**]. Robonaut is designed for research on human-robot collaboration in space, and human-like tasks. For more information, follow **this link**[1] to a talk on ROS in space from ROSCon 2014.

Being the first robot with ROS to be launched into space,

Example of an industrial robot with ROS goes here!   TODO!!!!!!!!!!

---

[1]https://vimeo.com/106993914

2.5   Software

2.5.1   Qt

2.5.2   PCL

2.6   The Kinect Sensor

2.7   Software Tools

2.7.1   Point Cloud Library

2.7.2   ROS

2.7.3   Qt

2.7.4   Current Research and Applications

2.8   Introduction to Sensors in Autonomous Robots

2.8.1   Depth Cameras

Different Methods for Depth Perception

In the context of this thesis, a depth camera is considered to be a sensor which the functionality of a regular video camera

A depth camera can be described as a regular color video camera with the ability to create spatial images. In the context of this thesis, a depth camera can more precisely be described as a RGB-D camera, which is short for red, green, blue and depth camera. A regular RGB camera will project a spatial scene onto a rectangular pixel grid, where each pixel contains intensity values for red, green and blue colors. These pixel values represents the detected scene. A major problem with RGB cameras is the significant loss of information. The information loss is mostly a consequence of 3d to 2d projection and digital quantization. RGB-D cameras have the means to reduce this information loss by mapping the pixel values to spatial coordinates, turning each pixel into voxels and the image into a point cloud of voxels.

Different variations of depth cameras will usually fall into one of two categories: active or passive. Passive sensors perceive the surroundings as it is, without actively interfering with the environment as a part of the sensing process. A typical passive RGB-D sensor is the stereo camera. Stereo cameras use a stream of synchronized image pairs to perceive depth. The image pairs are displaced along the horizontal axis, and the depth information is extracted by searching for mutual information in

the image pairs. How far the information is displaced from the left to the right image is directly related to how far away from the camera the information source is located.

Active sensors depend on some form of projection onto the surroundings. For depth cameras, the projection is usually in the form of laser or infra red light. In RGB-D cameras it is essential that the projected light is distinguishable from the visible spectrum. The Kinect sensor used in this project is an example of an active RGB-D sensor. A proper introduction to the Kinect, will follow shortly.

Natural User Interfaces - Origin of the Kinect

Forslag 1: When a group of designers are developing a new Graphical User Interface (GUI), they will often use a conceptual model when planning their design. The conceptual model is the mental model the designers want to put into the head of the user. All users will develop their own individual mental model, which is their high level understanding of how the GUI works. A conceptual model may contain metaphors for things the user already is familiar with. A painting program for example may use a metaphor for a canvas, paint brushes and palettes. When the mouse icon changes to a paint brush, most users will have an intuitive understanding of what it can do and how it is used. A Natural user interface (NUI) will seek to remove the metaphors and create a more seamless interaction between the user and the machine. Some NUIs may allow a user to write text with a pen instead of a keyboard, or dictate a letter with their voice while the computer converts audio into text.

Forslag 2: The idea behind a NUI is to make Human-Machine Interaction (HMI) as seamless and natural as possible. A NUI allows the user to communicate without tools such as a keyboard or a mouse. From starting as ideas, science fiction and rare research projects, NUIs can now be considered to be ubiquitous. Today, the most common form of NUIs is the touch screen found in smart phones and tablets.

The Microsoft Kinect sensor was initially designed as a NUI for the Xbox 360 gaming console. The sensor allows users to use gestures and sounds to play console games. Later on, Microsoft has released SDKs, enabling developers to create NUI applications for for Windows.

The modern RGB-D sensors which are commonly used in robot research projects today were initially intended as NUI.

Kinect for Xbox 360

Kinect for Xbox 360 is the RGB-D sensor used in this project. The device was initially intended as a NUI for gaming and office applications. Possible use cases

were inspired by early NUI research at Massachusetts Institute of Technology (MIT) and, later on, the science fiction movie Minority Report, where Tom Cruice interacts with a computer by using hand gestures [WA12]. The Kinect sensor is equipped with a depth sensor, a regular color camera, a microphone array and a tilt motor. The color camera in combination with the depth sensor forms what is usually referred to as a rgb-d sensor, i.e. a combined color and depth camera. This feature, combined with the relaticely low cost and accessability of the sensor as contributed to make the Kinect a very popular in research projects related to Simultanious Localization And Mapping (SLAM) and robotics.

Today, the the Kinect for Xbox 360 has been succeded by the Kinect for Xbox One, and is now considered to be a legacy device. Those considering to use the legacy Kinect should be aware of that it is becoming increasingly difficult, if not already impossible, to get hold of a new Kinect for Xbox 360.

2.8.2 Plannar Laser Sensors (LIDAR)

A plannar laser sensor, also known as e.g. laser proximity sensors or laser radars, can all be referred to as LIDARs.

Scanning Laser Range Finder, URG-04LX-UG01

2.8.3 Odometers

2.8.4 Sensor Fusion

2.9 RTAB-Map

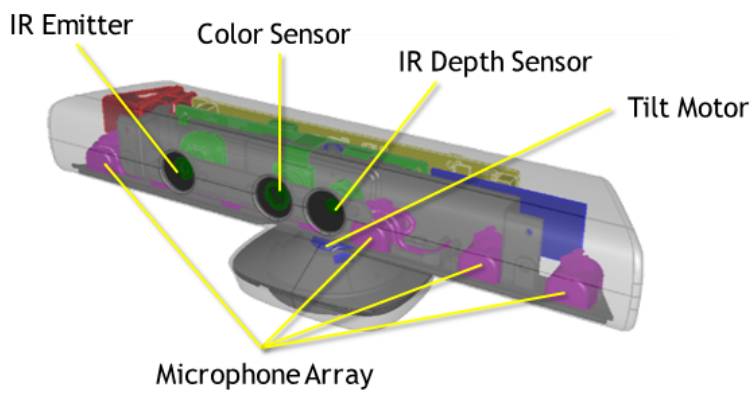**RTAB-map!** (**RTAB-map!**) [LM14].

Figure 2.1: Awesome Image

Figure 2.2: Awesome Image

Implementation

Implementation procedure for mobile robot:

- – Decide on ROS message interface.

- – Write interfaces for the motor drivers.

- – Create a description of the physical structure and properties of the robot in Unified Robot Description Format (URDF).

- – Extend the model to enable simulation in Gazebo.

- – Publish coordinate transform data via *tf* and visualize it in rviz.

- – Add sensors, with driver and simulation support.

- – Apply algorithms for navigation and other functionality.

3.1   Modelling

3.1.1   Physical Dimensions

The inertia tensor:

$$
I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}
\tag{3.1}
$$

Inertia tensor for a solid, uniform cylinder where the radius r is measured in parallel to the $x - y$ plane, and h is parallel to the z axis:

$$I_{cylinder} = \frac{1}{12}m \begin{bmatrix} (3r^2 + h^2) & 0 & 0 \\ 0 & (3r^2 + h^2) & 0 \\ 0 & 0 & r^2 \end{bmatrix} \tag{3.2}$$

Inertia tensor for a solid, uniform cuboid. The subscript of l indicates which axis l is measured along:

$$I_{cuboid} = \frac{1}{12}m \begin{bmatrix} (l_y^2 + l_z^2) & 0 & 0 \\ 0 & (l_x^2 + l_z^2) & 0 \\ 0 & 0 & (l_x^2 + l_y^2) \end{bmatrix} \tag{3.3}$$

3.1.2   Coordinate Frames

3.2   Simulations

3.3   ROS Nodes for Motion Control

3.3.1

3.4   Operator Control Station (OCS)

The Operator Control Station (OCS) allows an operator to control and monitor the robot through a graphical user interface.

3.4.1   Graphical User Interface

A Qt-based GUI...

3.5   The Handheld Remote Control - "Robot Leash"

http://developer.samsung.com/technical-doc/view.do?v=T000000117

Chapter 4

Testing

4.1   Testplan

4.2   Results

4.3   Discussion

Chapter 5

Discussion

Chapter 6
Conclusion

6.1 Future Work

6.1.1 Autonomous Non-Destructive Testing

Advancements in Artificial Intelligence (AI), big data and machine learning opens up exciting possibilities for autonomous Non-destructive Testing (NDT). Branches of this technology is usually encountered in the context of image recognition, i.e. teaching machines to understand what they see. The same concepts may be applied to forms of NDT besides regular visual sensor input, such as ultra sound or eddy currents for corrosion detection.

6.1.2 Large Scale Kinect Fusion - Kintinous

Kinect Fusion has great potential for augmented reality. Augmented reality is a concept which blends the real and virtual environment. This opens up opportunities to create realistic and immersive training scenarios for the operators. Unfortunately, Kinect Fusion is limited reconstructing a rather small volume depending on the resolution. By varying the resolution, volumes can at the least cover a normal office desk and at the most cover a small room [**?**].

Kintinous...

A guide on how to build Kintinous can be found at https://github.com/mp3guy/Kintinuous. The procedure is complicated, as it usually is for experimental builds. It is recommended to attempt the procedure on a fresh install of Ubuntu 14.04 or 15.04 [Kin].

6.2   Task Fulfillment

6.3   Task Fulfilment

6.4   Final Conclusion

Chapter 7

Setting Up the Project

7.1 Installation

7.1.1 Equipment List

Item List

Software list

Hector SLAM for ROS Install with

```
sudo apt-get install ros-indigo-hector-slam
```

Compatibility Issues

Indigo, Ubuntu etc.

7.1.2 Install Ubuntu

7.1.3 Download ROS

7.2 Configuring the Project

7.2.1 Configuring the ROS Workspace

7.2.2 Configuring the Bluetooth Connection

The Qt framework is used to simplify the implementation of the Bluetooth connection between the ROS graph and a remote device. Our ROS installation for this project already includes some variant of Qt version 4.8. While useful for creating new GUI applications, it lacks a Bluetooth API. The latest version of Qt, version 5.5, is equipped with libraries necessary for developing Bluetooth applications. This part of

the guide explain how to create a Qt 5 application which can be build by *catkin_make* and run as a *rosnode*.

7.2.3   Adding the Qwt Plugin For Customizing QDial

This is a "howto" guide on how to install the Qwt-plugin.The following is assumed:

– Development is performed on Windows 7

– Qt 5.x is installed at c:/Qt

Chapter 8
Troubleshooting

8.1 Introduction

This part of the thesis contains answers to some of the problems that was encountered over the course od the semester. The solutions are not complete or comprehensive, but may provide some quick fixes for any students that may continue working with this project.

8.2 Gazebo

8.2.1 Error [Node.cc.90] No namespace found

Solution: Remember to source the *gazebo* installation. In this case, with *gazebo-2.2* installed as recommended for ROS Indigo, the setup file can be sourced by entering

```
$ source /usr/share/gazebo-2.2/setup.sh
```

8.2.2 Dependency Issues When Installing *gazebo2*

This problem was encountered after removing gazebo and then typing

```
$ sudo apt-get upgrade
```

When typing

```
$ sudo apt-get install -y gazebo2
```

the installation failed because some dependencies had been upgraded to an incompatible version. To solve this, take note of the missing dependencies listed after

entering the command above, open Ubuntu Software Center and select the History tab. Scroll down and locate the missing dependencies. They should have a red X next to them, indicating that they have been uninstalled. Then, enter the following command:

```
$ sudo apt-get install <NAME OF THE UNINSTALLED DEPENDENCY>
```

gazebo2

8.3   Ubuntu

8.3.1   Ubuntu Freezes

Sometimes during work with the project, Ubuntu would freeze and become unresponsive to keyboard input and mouse clicks. The mouse could be moved around, but was otherwise unresponsive. This event occurred exclusively when using *RVIZ* and displaying a camera topic as an image in the lower left corner of the GUI. The following steps from a post at askubuntu.com, solves the problem [Reb]:

While holding ⟨Alt⟩ and ⟨SysReq (Print Screen)⟩ , type ⟨R⟩⟨E⟩⟨I⟩⟨S⟩⟨U⟩⟨B⟩ . Press each key properly, and allow a few seconds to pass between each keystroke so that each command has time to execute. This should cause the computer to reboot, and is supposedly safer than using the power button. Follow **this** link to the post for more details.

References

[DRC]   Darpa robotics challenge. http://www.theroboticschallenge.org/overview. Accessed: 2016-04-05.

[ER12]   Mohamed A. El-Reedy. Chapter 6 - corrosion protection. In Mohamed A. El-Reedy, editor, *Offshore Structures*, pages 383 – 443. Gulf Professional Publishing, Boston, 2012.

[ifr]   International federation of robotics - statistics. http://www.ifr.org/industrial-robots/statistics/. Accessed: 2016-04-05.

[Kin]   Kintinous. https://github.com/mp3guy/Kintinuous. Accessed: 2016-03-21.

[LM14]   M. Labbé and F. Michaud. Online global loop closure detection for large-scale multi-session graph-based slam. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 2661–2666, Sept 2014.

[QGS15]   Morgan Quigley, Brian Gerkey, and William D. Smart. *Programming Robots with ROS*. O'Reilly Media, Inc., December 2015.

[Reb]   Reboot ubuntu. http://askubuntu.com/questions/4408/what-should-i-do-when-ubuntu-freezes/36717#36717. Accessed: 2016-03-14.

[ROSa]   ROS history. http://www.ros.org/history/. Accessed: 2016-02-28.

[ROSb]   Ros-industrial. http://rosindustrial.org/the-challenge/. Accessed: 2016-04-05.

[ROSc]   ROS installation. http://wiki.ros.org/indigo/Installation/Ubuntu. Accessed: 2016-02-29.

[SS14]   Thomas Barz Andreas Kroll Dirk Schulz Samuel Soldan, Jochen Welle. Towards autonomous robotic systems for remote gas leak detection and localization in industrial environments. In *Field and Service Robotics*, pages 233–247. Springer Berlin Heidelberg, 2014.

[WA12]   Jarrett Webb and James Ashley. *Beginning Kinect Programming with the Microsoft Kinect SDK*. Apress, 2012.